Inapproximability of maximal strip recovery[*]

Minghui Jiang

Department of Computer Science, Utah State University, Logan, UT 84322, USA
mjiang@cc.usu.edu

June 7, 2010

Abstract

In comparative genomic, the first step of sequence analysis is usually to decompose two or more genomes into syntenic blocks that are segments of homologous chromosomes. For the reliable recovery of syntenic blocks, noise and ambiguities in the genomic maps need to be removed first. Maximal Strip Recovery (MSR) is an optimization problem proposed by Zheng, Zhu, and Sankoff for reliably recovering syntenic blocks from genomic maps in the midst of noise and ambiguities. Given d genomic maps as sequences of gene markers, the objective of MSR-d is to find d subsequences, one subsequence of each genomic map, such that the total length of syntenic blocks in these subsequences is maximized. For any constant $d \geq 2$, a polynomial-time $2d$-approximation for MSR-d was previously known. In this paper, we show that for any $d \geq 2$, MSR-d is APX-hard, even for the most basic version of the problem in which all gene markers are distinct and appear in positive orientation in each genomic map. Moreover, we provide the first explicit lower bounds on approximating MSR-d for all $d \geq 2$. In particular, we show that MSR-d is NP-hard to approximate within $\Omega(d/\log d)$. From the other direction, we show that the previous $2d$-approximation for MSR-d can be optimized into a polynomial-time algorithm even if d is not a constant but is part of the input. We then extend our inapproximability results to several related problems including CMSR-d, δ-gap-MSR-d, and δ-gap-CMSR-d.

Keywords: computational complexity, bioinformatics, sequence analysis, genome rearrangement.

1 Introduction

In comparative genomic, the first step of sequence analysis is usually to decompose two or more genomes into syntenic blocks that are segments of homologous chromosomes. For the reliable recovery of syntenic blocks, noise and ambiguities in the genomic maps need to be removed first. A genomic map is a sequence of gene markers. A gene marker appears in a genomic map in either positive or negative orientation. Given d genomic maps, *Maximal Strip Recovery* (MSR-d) is the problem of finding d subsequences, one subsequence of each genomic map, such that the total length of strips of these subsequences is maximized [27, 11]. Here a *strip* is a maximal string of at least two markers such that either the string itself or its signed reversal appears contiguously as a substring in each of the d subsequences in the solution. Without loss of generality, we can assume that all markers appear in positive orientation in the first genomic map.

For example, the two genomic maps (the markers in negative orientation are underlined)

$$\begin{array}{cccccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ \underline{8} & \underline{5} & \underline{7} & \underline{6} & 4 & 1 & 3 & 2 & \underline{12} & \underline{11} & \underline{10} & 9 \end{array}$$

[*]This research was supported in part by NSF grant DBI-0743670. A preliminary version of this paper appeared in two parts [17, 18] in the Proceedings of the 20th International Symposium on Algorithms and Computation (ISAAC 2009) and the Proceedings of the 4th International Frontiers of Algorithmics Workshop (FAW 2010).

have two subsequences

$$\begin{array}{ccccccccc}
1 & 3 & & 6 & 7 & 8 & 10 & 11 & 12 \\
\underline{8} & \underline{7} & \underline{6} & & 1 & 3 & \underline{12} & \underline{11} & \underline{10}
\end{array}$$

of the maximum total strip length 8. The strip $\langle 1, 3\rangle$ is positive and forward in both subsequences; the other two strips $\langle 6, 7, 8\rangle$ and $\langle 10, 11, 12\rangle$ are positive and forward in the first subsequence, but are negative and backward in the second subsequence. Intuitively, the strips are syntenic blocks, and the deleted markers not in the strips are noise and ambiguities in the genomic maps.

The problem MSR-2 was introduced by Zheng, Zhu, and Sankoff [27], and was later generalized to MSR-d for any $d \geq 2$ by Chen, Fu, Jiang, and Zhu [11]. For MSR-2, Zheng et al. [27] presented a potentially exponential-time heuristic that solves a subproblem of Maximum-Weight Clique. For MSR-d, Chen et al. [11] presented a $2d$-approximation based on Bar-Yehuda et al.'s fractional local-ratio algorithm for Maximum-Weight Independent Set in d-interval graphs [6]; the running time of this $2d$-approximation algorithm is polynomial if d is a constant.

On the complexity side, Chen et al. [11] showed that several close variants of the problem MSR-d are intractable. In particular, they showed that (i) MSR-2 is NP-complete if duplicate markers are allowed in each genomic map, and that (ii) MSR-3 is NP-complete even if the markers in each genomic map are distinct. The complexity of MSR-2 with no duplicates, however, was left as an open problem.

In the biological context, a genomic map may contain duplicate markers as a paralogy set [27, p. 516], but such maps are relatively rare. Thus MSR-2 without duplicates is the most useful version of MSR-d in practice. Theoretically, MSR-2 without duplicates is the most basic and hence the most interesting version of MSR-d. Also, the previous NP-hardness proofs of both (i) MSR-2 with duplicates and (ii) MSR-3 without duplicates [11] rely on the fact that a marker may appear in a genomic map in either positive or negative orientation. A natural question is whether there is any version of MSR-d that remains NP-hard even if all markers in the genomic maps are in positive orientation.

We give a precise formulation of *the most basic version* of the problem MSR-d as follows:

INSTANCE: Given d sequences G_i, $1 \leq i \leq d$, where each sequence is a permutation of $\langle 1, \ldots, n\rangle$.

QUESTION: Find a subsequence G'_i of each sequence G_i, $1 \leq i \leq d$, and find a set of strips S_j, where each strip is a sequence of length at least two over the alphabet $\{1, \ldots n\}$, such that each subsequence G'_i is the concatenation of the strips S_j in some order, and the total length of the strips S_j is maximized.

The main result of this paper is the following theorem that settles the computational complexity of the most basic version of Maximal Strip Recovery, and moreover provides the first explicit lower bounds on approximating MSR-d for all $d \geq 2$:

Theorem 1. MSR-d *for any* $d \geq 2$ *is APX-hard. Moreover,* MSR-2, MSR-3, MSR-4, *and* MSR-d *are NP-hard to approximate within* 1.000431, 1.002114, 1.010661, *and* $\Omega(d/\log d)$, *respectively, even if all markers are distinct and appear in positive orientation in each genomic map.*

Recall that for any constant $d \geq 2$, MSR-d admits a polynomial-time $2d$-approximation algorithm [11]. Thus MSR-d for any constant $d \geq 2$ is APX-complete. Our following theorem gives a polynomial-time $2d$-approximation algorithm for MSR-d even if the number d of genomic maps is not a constant but is part of the input:

Theorem 2. *For any* $d \geq 2$*, there is a polynomial-time* $2d$*-approximation algorithm for* MSR-d *if all markers are distinct in each genomic map. This holds even if* d *is not a constant but is part of the input.*

Compare the upper bound of $2d$ in Theorem 2 and the asymptotic lower bound of $\Omega(d/\log d)$ in Theorem 1.

Maximal Strip Recovery [27, 11] is a maximization problem. Wang and Zhu [26] introduced Complement Maximal Strip Recovery as a minimization problem. Given d genomic maps as input, the problem CMSR-d is the same as the problem MSR-d except that the objective is minimizing the number of deleted markers not in the strips, instead of maximizing the number of markers in the strips. A natural question is whether a polynomial-time approximation scheme may be obtained for this problem. Our following theorem shows that unless NP $=$ P, CMSR-d cannot be approximated arbitrarily well:

Theorem 3. CMSR-d *for any* $d \geq 2$ *is APX-hard. Moreover,* CMSR-2, CMSR-3, CMSR-4, *and* CMSR-d *for any* $d \geq 173$ *are NP-hard to approximate within* 1.000625, 1.0101215, 1.0202429, *and* $\frac{7}{6} - O(\log d/d)$, *respectively, even if all markers are distinct and appear in positive orientation in each genomic map. If the number* d *of genomic maps is not a constant but is part of the input, then* CMSR-d *is NP-hard to approximate within any constant less than* $10\sqrt{5}-21 = 1.3606\ldots$, *even if all markers are distinct and appear in positive orientation in each genomic map.*

Note the similarity between Theorem 1 and Theorem 3. In fact, our proof of Theorem 3 uses exactly the same constructions as our proof of Theorem 1. The only difference is in the analysis of the approximation lower bounds.

Bulteau, Fertin, and Rusu [10] recently proposed a restricted variant of Maximal Strip Recovery called δ-gap-MSR, which is MSR-2 with the additional constraint that at most δ markers may be deleted between any two adjacent markers of a strip in each genomic map. We now define δ-gap-MSR-d and δ-gap-CMSR-d as the restricted variants of the two problems MSR-d and CMSR-d, respectively, with the additional δ-gap constraint. Bulteau et al. [10] proved that δ-gap-MSR-2 is APX-hard for any $\delta \geq 2$, and is NP-hard for $\delta = 1$. We extend our proofs of Theorem 1 and Theorem 3 to obtain the following theorem on δ-gap-MSR-d and δ-gap-CMSR-d for any $\delta \geq 2$:

Theorem 4. *Let* $\delta \geq 2$. *Then*

(1) δ-gap-MSR-d *for any* $d \geq 2$ *is APX-hard. Moreover,* δ-gap-MSR-2, δ-gap-MSR-3, δ-gap-MSR-4, *and* δ-gap-MSR-d *are NP-hard to approximate within* 1.000431, 1.002114, 1.010661, *and* $d/2^{O(\sqrt{\log d})}$, *respectively, even if all markers are distinct and appear in positive orientation in each genomic map.*

(2) δ-gap-CMSR-d *for any* $d \geq 2$ *is APX-hard. Moreover,* δ-gap-CMSR-2, δ-gap-CMSR-3, δ-gap-CMSR-4, *and* δ-gap-CMSR-d *for any* $d \geq 173$ *are NP-hard to approximate within* 1.000625, 1.0101215, 1.0202429, *and* $\frac{7}{6} - O(\log d/d)$, *respectively, even if all markers are distinct and appear in positive orientation in each genomic map. If the number* d *of genomic maps is not a constant but is part of the input, then* δ-gap-CMSR-d *is NP-hard to approximate within any constant less than* $10\sqrt{5} - 21 = 1.3606\ldots$, *even if all markers are distinct and appear in positive orientation in each genomic map.*

We refer to [13, 20, 9] for some related results. Maximal Strip Recovery is a typical combinatorial problem in biological sequence analysis, in particular, genome rearrangement. The earliest inapproximability result for genome rearrangement problems is due to Berman and Karpinski [7], who proved that Sorting by Reversals is NP-hard to approximate within any constant less than $\frac{1237}{1236}$. More recently, Zhu and Wang [28] proved that Translocation Distance is NP-hard to approximate within any constant less than $\frac{5717}{5716}$. Similar inapproximability results have also been obtained for other important problems in bioinformatics. For example, Nagashima and Yamazaki [23] proved that Non-overlapping Local Alignment is NP-hard to approximate within any constant less than $\frac{8668}{8665}$, and Manthey [22] proved that Multiple Sequence Alignment with weighted sum-of-pairs score is APX-hard for arbitrary metric scoring functions over the binary alphabet.

The rest of this paper is organized as follows. We first review some preliminaries in Section 2. Then, in Sections 3, 4, 5, and 6, we show that MSR-d for any $d \geq 2$ is APX-hard, and prove explicit approximation lower bounds. (For any two constants d and d' such that $d' > d \geq 2$, the problem MSR-d is a special case of the problem MSR-d' with $d' - d$ redundant genomic maps. Thus the APX-hardness of MSR-2 implies the APX-hardness of MSR-d for all constants $d \geq 2$. To present the ideas progressively, however, we show that MSR-4, MSR-3, and MSR-2 are APX-hard by three different L-reductions of increasing sophistication.) In Section 7, we present a $2d$-approximation algorithm for MSR-d that runs in polynomial time even if the number d of genomic maps is not a constant but is part of the input. In Section 8, we present inapproximability results for CMSR-d, δ-gap-MSR-d, and δ-gap-CMSR-d. We conclude with remarks in Section 9.

2 Preliminaries

L-reduction. Given two optimization problems X and Y, an *L-reduction* [24] from X to Y consists of two polynomial-time functions f and g and two positive constants α and β satisfying the following two properties:

1. For every instance x of X, $f(x)$ is an instance of Y such that
$$\mathrm{opt}(f(x)) \leq \alpha \cdot \mathrm{opt}(x), \tag{1}$$

2. For every feasible solution y to $f(x)$, $g(y)$ is a feasible solution to x such that
$$|\mathrm{opt}(x) - \mathrm{val}(g(y))| \leq \beta \cdot |\mathrm{opt}(f(x)) - \mathrm{val}(y)|. \tag{2}$$

Here $\mathrm{opt}(x)$ denotes the value of the optimal solution to an instance x, and $\mathrm{val}(y)$ denotes the value of a solution y. The two properties of L-reduction imply the following inequality on the relative errors of approximation:
$$\frac{|\mathrm{opt}(x) - \mathrm{val}(g(y))|}{\mathrm{opt}(x)} \leq \alpha\beta \cdot \frac{|\mathrm{opt}(f(x)) - \mathrm{val}(y)|}{\mathrm{opt}(f(x))}.$$

A relative error of ϵ corresponds to an approximation factor of $1 + \epsilon$ for a minimization problem, and corresponds to an approximation factor of $\frac{1}{1-\epsilon}$ for a maximization problem. Thus we have the following propositions:

1. For a minimization problem X and a minimization problem Y, if X is NP-hard to approximate within $1 + \alpha\beta\epsilon$, then Y is NP-hard to approximate within $1 + \epsilon$.

2. For a maximization problem X and a maximization problem Y, if X is NP-hard to approximate within $\frac{1}{1-\alpha\beta\epsilon}$, then Y is NP-hard to approximate within $\frac{1}{1-\epsilon}$.

3. For a minimization problem X and a maximization problem Y, if X is NP-hard to approximate within $1 + \alpha\beta\epsilon$, then Y is NP-hard to approximate within $\frac{1}{1-\epsilon}$.

4. For a maximization problem X and a minimization problem Y, if X is NP-hard to approximate within $\frac{1}{1-\alpha\beta\epsilon}$, then Y is NP-hard to approximate within $1 + \epsilon$.

APX-hard optimization problems. We review the complexities of some APX-hard optimization problems that will be used in our reductions.

- Max-IS-Δ is the problem Maximum Independent Set in graphs of maximum degree Δ. Max-IS-3 is APX-hard; see [4]. Moreover, Chlebík and Chlebíková [12] showed that Max-IS-3 and Max-IS-4 are NP-hard to approximate within 1.010661 and 1.0215517, respectively. Trevisan [25] showed that Max-IS-Δ is NP-hard to approximate within $\Delta/2^{O(\sqrt{\log \Delta})}$.
- Min-VC-Δ is the problem Minimum Vertex Cover in graphs of maximum degree Δ. Min-VC-3 is APX-hard; see [4]. Moreover, Chlebík and Chlebíková [12] showed that Min-VC-3 and Min-VC-4 are NP-hard to approximate within 1.0101215 and 1.0202429, respectively, and, for any $\Delta \geq 228$, Min-VC-Δ is NP-hard to approximate within $\frac{7}{6} - O(\log \Delta/\Delta)$. Dinur and Safra [14] showed that Minimum Vertex Cover is NP-hard to approximate within any constant less than $10\sqrt{5} - 21 = 1.3606\ldots$.
- Given a set X of n variables and a set $\mathcal{C}$ of m clauses, where each variable has exactly p literals (in p different clauses) and each clause is the disjunction of exactly q literals (of q different variables), Ep-Occ-Max-Eq-SAT is the problem of finding an assignment of X that satisfies the maximum number of clauses in $\mathcal{C}$. Note that $np = mq$. Berman and Karpinski [8] showed that E3-Occ-Max-E2-SAT is NP-hard to approximate within any constant less than $\frac{464}{463}$.
- Given d disjoint sets V_i of vertices, $1 \leq i \leq d$, and given a set $E \subseteq V_1 \times \cdots \times V_d$ of hyper-edges, d-Dimensional-Matching is the problem of finding a maximum-cardinality subset $M \subseteq E$ of pairwise-disjoint hyper-edges. Hazan, Safra, and Schwartz [16] showed that d-Dimensional-Matching is NP-hard to approximate within $\Omega(d/\log d)$.

Linear forest and linear arboricity. A *linear forest* is a graph in which every connected component is a path. The *linear arboricity* of a graph is the minimum number of linear forests into which the edges of the graph can be decomposed. Akiyama, Exoo, and Harary [2, 3] conjectured that the linear arboricity of every graph G of maximum degree Δ satisfies $\mathrm{la}(G) \leq \lceil(\Delta+1)/2\rceil$. This conjecture has been confirmed for graphs of small constant degrees, and has been shown to be asymptotically correct as $\Delta \to \infty$ [5]. In particular, the proof of the conjecture for $\Delta = 3$ and 4 are constructive [2, 1, 3] and lead to polynomial-time algorithms for decomposing any graph of maximum degree $\Delta = 3$ and 4 into at most $\lceil(\Delta+1)/2\rceil = 2$ and 3 linear forests, respectively. Also, the proof of the first upper bound on linear arboricity by Akiyama, Exoo, and Harary [3] implies a simple polynomial-time algorithm for decomposing any graph of maximum degree Δ into at most $\lceil 3\lceil\Delta/2\rceil/2\rceil$ linear forests.

Define

$$f(\Delta) = \max_G f(G),$$

where G ranges over all graphs of maximum degree Δ, and $f(G)$ denotes the number of linear forests that Akiyama, Exoo, and Harary's algorithm [3] decomposes G into. Then

$$\lceil(\Delta+1)/2\rceil \leq f(\Delta) \leq \lceil 3\lceil\Delta/2\rceil/2\rceil. \tag{3}$$

3 MSR-4 is APX-hard

In this section, we prove that MSR-4 is APX-hard by a simple L-reduction from Max-IS-3. Before we present the L-reduction, we first show that MSR-4 is NP-hard by a reduction in the classical style, which is perhaps more familiar to most readers. Throughout this paper, we follow this progressive format of presentation.

3.1 NP-hardness reduction from Max-IS-3 to MSR-4

Let G be a graph of maximum degree 3. Let n be the number of vertices in G. Partition the edges of G into two linear forests E_1 and E_2. Let V_1 and V_2 be the vertices of G that are *not* incident to any edges in E_1 and in E_2, respectively. We construct four genomic maps $G_\rightarrow$, $G_\leftarrow$, G_1, and G_2, where each map is a permutation of the following $2n$ distinct markers all in positive orientation:

- n pairs of vertex markers $\overset{i}{\subset}$ and $\overset{i}{\supset}$, $1 \le i \le n$.

$G_\rightarrow$ and $G_\leftarrow$ are concatenations of the n pairs of vertex markers with ascending and descending indices, respectively:

$$\begin{array}{rccc} G_\rightarrow: & \overset{1}{\subset}\ \overset{1}{\supset} & \cdots & \overset{n}{\subset}\ \overset{n}{\supset} \\ G_\leftarrow: & \overset{n}{\subset}\ \overset{n}{\supset} & \cdots & \overset{1}{\subset}\ \overset{1}{\supset} \end{array}$$

G_1 and G_2 are represented schematically as follows:

$$\begin{array}{rcc} G_1: & \langle E_1 \rangle & \langle V_1 \rangle \\ G_2: & \langle E_2 \rangle & \langle V_2 \rangle \end{array}$$

$\langle E_1 \rangle$ and $\langle E_2 \rangle$ consist of vertex markers of the vertices incident to the edges in E_1 and E_2, respectively. The markers of the vertices in each path $v_1 v_2 \ldots v_k$ are grouped together in an interleaving pattern: for $1 \le i \le k$, the left marker of v_i, the right marker of v_{i-1} (if $i > 1$), the left marker of v_{i+1} (if $i < k$), and the right marker of v_i are consecutive.

$\langle V_1 \rangle$ and $\langle V_2 \rangle$ consist of vertex markers of the vertices in V_1 and V_2, respectively. The left marker and the right marker of each pair are consecutive.

This completes the construction. We refer to Figure 1 (a) and (b) for an example.

Two pairs of markers *intersect* in a genomic map if a marker of one pair appears between the two markers of the other pair. The following property of our construction is obvious:

Proposition 1. *Two vertices are adjacent in the graph G if and only if the corresponding two pairs of vertex markers intersect in one of the two genomic maps G_1, G_2.*

We say that four subsequences of the four genomic maps $G_\rightarrow, G_\leftarrow, G_1, G_2$ are *canonical* if each strip of the subsequences is a pair of vertex markers. We have the following lemma on canonical subsequences:

Lemma 1. *In any four subsequences of the four genomic maps $G_\rightarrow, G_\leftarrow, G_1, G_2$, respectively, each strip must be a pair of vertex markers.*

Proof. By construction, a strip cannot include two vertex markers of different indices because they appear in different orders in $G_\rightarrow$ and in $G_\leftarrow$. □

The following lemma establishes the NP-hardness of MSR-4:

Lemma 2. *The graph G has an independent set of at least k vertices if and only if the four genomic maps $G_\rightarrow, G_\leftarrow, G_1, G_2$ have four subsequences whose total strip length l is at least $2k$.*

Proof. We first prove the "only if" direction. Suppose that the graph G has an independent set of at least k vertices. We will show that the four genomic maps $G_\rightarrow, G_\leftarrow, G_1, G_2$ have four subsequences of total strip length at least $2k$. By Proposition 1, the k vertices in the independent set correspond to k pairs of vertex markers that do not intersect each other in the genomic maps. These k pairs of vertex markers induce a subsequence of length $2k$ in each genomic map. In each subsequence, the left marker and the right marker

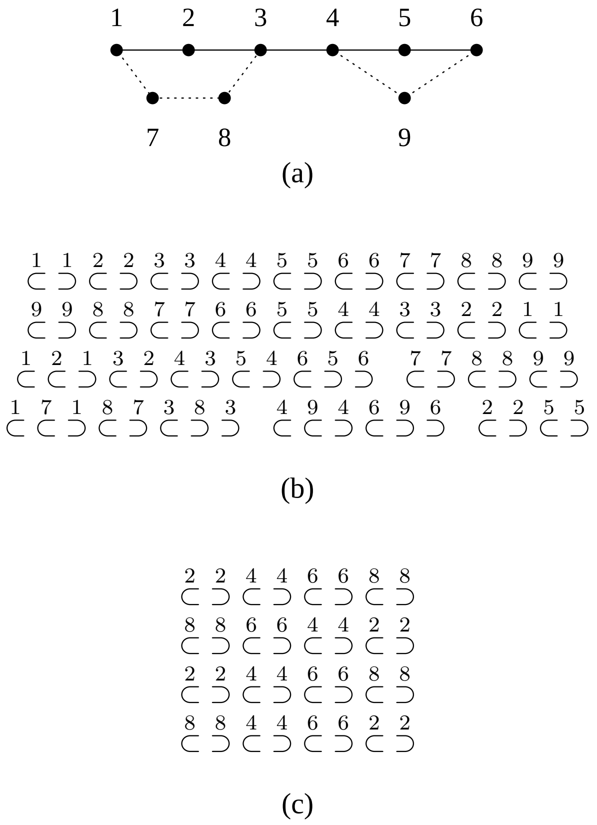

Figure 1: (a) The graph G: E_1 is a single solid path $\langle 1, 2, 3, 4, 5, 6 \rangle$, E_2 consists of two dotted paths $\langle 1, 7, 8, 3 \rangle$ and $\langle 4, 9, 6 \rangle$, $V_1 = \{7, 8, 9\}$, $V_2 = \{2, 5\}$. (b) The four genomic maps $G_{\rightarrow}, G_{\leftarrow}, G_1, G_2$. (c) The four subsequences of the genomic maps corresponding to the independent set $\{2, 4, 6, 8\}$ in the graph.

of each pair appear consecutively and compose a strip. Thus the total strip length is at least $2k$. We refer to Figure 1(c) for an example.

We next prove the "if" direction. Suppose that the four genomic maps $G_{\rightarrow}, G_{\leftarrow}, G_1, G_2$ have four subsequences of total strip length at least $2k$. We will show that the graph G has an independent set of at least k vertices. By Lemma 1, each strip of the subsequences must be a pair of vertex markers. Thus we obtain at least k pairs of vertex markers that do not intersect each other in the genomic maps. Then, by Proposition 1, the corresponding set of at least k vertices in the graph G form an independent set. □

3.2 L-reduction from Max-IS-3 to MSR-4

We present an L-reduction (f, g, α, β) from Max-IS-3 to MSR-4 as follows. The function f, given a graph G of maximum degree 3, constructs the four genomic maps $G_{\rightarrow}, G_{\leftarrow}, G_1, G_2$ as in the NP-hardness reduction. Let k^* be the number of vertices in a maximum independent set in G, and let l^* be the maximum total strip length of any four subsequences of $G_{\rightarrow}, G_{\leftarrow}, G_1, G_2$, respectively. By Lemma 2, we have

$$l^* = 2k^*.$$

Choose $\alpha = 2$, then property (1) of L-reduction is satisfied.

The function g, given four subsequences of the four genomic maps $G_{\rightarrow}, G_{\leftarrow}, G_1, G_2$, respectively, returns an independent set of vertices in the graph G corresponding to the pairs of vertex markers that are strips of the subsequences. Let l be the total strip length of the subsequences, and let k be the number of vertices in the independent set returned by the function g. Then $k \geq l/2$. It follows that

$$|k^* - k| = k^* - k \leq l^*/2 - l/2 = |l^* - l|/2.$$

Choose $\beta = 1/2$, then property (2) of L-reduction is also satisfied.

We have obtained an L-reduction from Max-IS-3 to MSR-4 with $\alpha\beta = 1$. Chlebík and Chlebíková [12] showed that Max-IS-3 is NP-hard to approximate within 1.010661. It follows that MSR-4 is also NP-hard to approximate within 1.010661. The lower bound extends to MSR-d for all constants $d \geq 4$.

The L-reduction from Max-IS-3 to MSR-4 can be obviously generalized:

Lemma 3. *Let $\Delta \geq 3$ and $d \geq 4$. If there is a polynomial-time algorithm for decomposing any graph of maximum degree Δ into $d - 2$ linear forests, then there is an L-reduction from* Max-IS-Δ *to* MSR-d *with constants $\alpha = 2$ and $\beta = 1/2$.*

4 MSR-3 is APX-hard

In this section, we prove that MSR-3 is APX-hard by a slightly more sophisticated L-reduction again from Max-IS-3.

4.1 NP-hardness reduction from Max-IS-3 to MSR-3

Let G be a graph of maximum degree 3. Let n be the number of vertices in G. Partition the edges of G into two linear forests E_1 and E_2. Let V_1 and V_2 be the vertices of G that are *not* incident to any edges in E_1 and E_2, respectively. We construct three genomic maps G_0, G_1, and G_2, where each map is a permutation of the following $4n$ distinct markers all in positive orientation:

- n pairs of vertex markers $\overset{i}{\subset}$ and $\overset{i}{\supset}$, $1 \leq i \leq n$;
- n pairs of dummy markers $\overset{i}{\sqsubset}$ and $\overset{i}{\sqsupset}$, $1 \leq i \leq n$.

G_0 consists of the $2n$ pairs of vertex and dummy markers in an alternating pattern:

$$\overset{1}{\subset}\ \overset{1}{\supset}\quad \overset{1}{\sqsubset}\ \overset{1}{\sqsupset}\quad \cdots\quad \overset{n}{\subset}\ \overset{n}{\supset}\quad \overset{n}{\sqsubset}\ \overset{n}{\sqsupset}$$

G_1 and G_2 are represented schematically as follows:

$$\begin{array}{llll} G_1: & \langle V_1\rangle & \langle E_1\rangle & \langle D\rangle \\ G_2: & \langle D\rangle & \langle E_2\rangle & \langle V_2\rangle \end{array}$$

$\langle E_1\rangle$ and $\langle E_2\rangle$ consist of vertex markers of the vertices incident to the edges in E_1 and E_2, respectively. The markers of the vertices in each path $v_1v_2 \ldots v_k$ are grouped together in an interleaving pattern: for $1 \leq i \leq k$, the left marker of v_i, the right marker of v_{i-1} (if $i > 1$), the left marker of v_{i+1} (if $i < k$), and the right marker of v_i are consecutive.

$\langle V_1\rangle$ and $\langle V_2\rangle$ consist of vertex markers of the vertices in V_1 and V_2, respectively. The left marker and the right marker of each pair are consecutive.

$\langle D\rangle$ is the reverse permutation of the n pairs of dummy markers:

$$\overset{n}{\sqsubset}\ \overset{n}{\sqsupset}\quad \cdots\quad \overset{1}{\sqsubset}\ \overset{1}{\sqsupset}$$

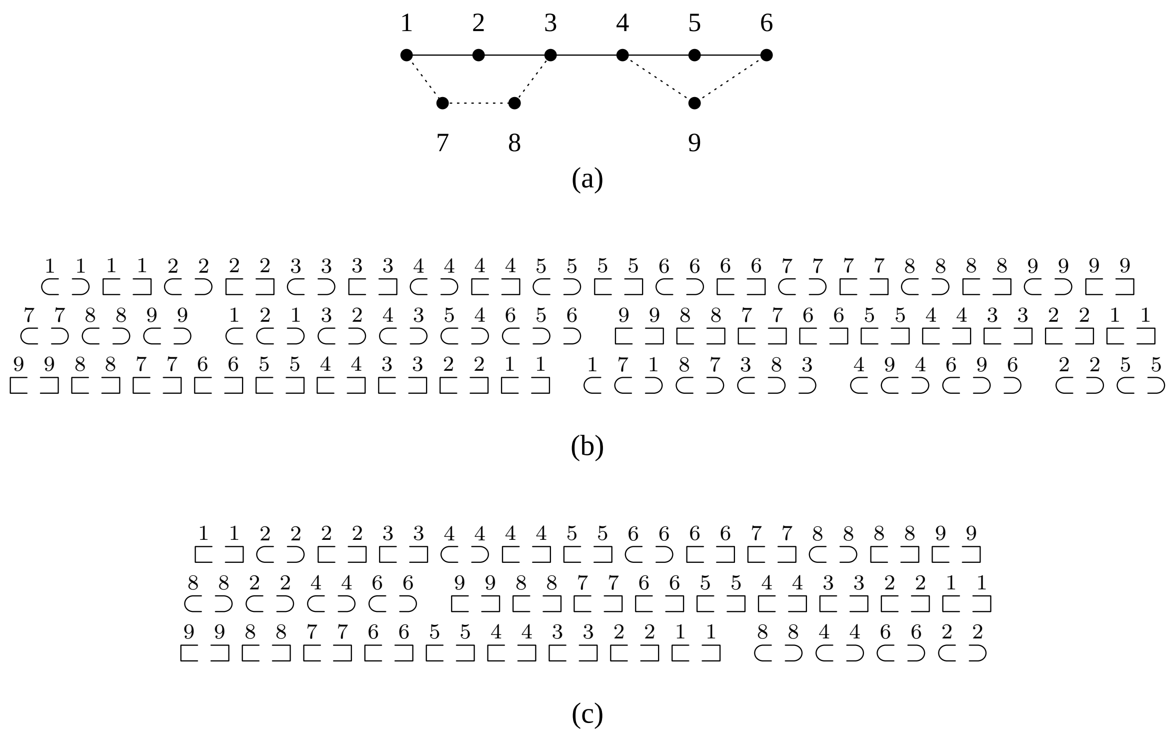

Figure 2: (a) The graph G: E_1 is a single (solid) path $\langle 1, 2, 3, 4, 5, 6\rangle$, E_2 consists of two (dotted) paths $\langle 1, 7, 8, 3\rangle$ and $\langle 4, 9, 6\rangle$, $V_1 = \{7, 8, 9\}$, $V_2 = \{2, 5\}$. (b) The three genomic maps G_0, G_1, G_2. (c) The three subsequences of the genomic maps corresponding to the independent set $\{2, 4, 6, 8\}$ in the graph.

This completes the construction. We refer to Figure 2 (a) and (b) for an example.

It is clear that Proposition 1 still holds. The following lemma on canonical subsequences is analogous to Lemma 1:

Lemma 4. *If the three genomic maps G_0, G_1, G_2 have three subsequences of total strip length l, then they must have three subsequences of total strip length at least l such that* (i) *each strip is either a pair of vertex markers or a pair of dummy markers, and* (ii) *each pair of dummy markers is a strip.*

Proof. We present an algorithm that transforms the subsequences into canonical form without reducing the total strip length. By construction, a strip cannot include both a dummy marker and a vertex marker because they appear in different orders in G_1 and in G_2, and a strip cannot include two dummy markers of different indices because they appear in different orders in G_0 and in G_1 and G_2. Suppose that a strip S consists of vertex markers of two or more different indices. Then there must be two vertex markers μ and ν of different indices i and j that are consecutive in S. Since the vertex markers and the dummy markers appear in G_0 in an alternating pattern with ascending indices, we must have $i < j$. Moreover, the pair of dummy markers of index i, which appears between μ and ν in G_0, must be missing from the subsequences. Now cut the strip S into S_μ and S_ν between μ and ν. If S_μ (resp. S_ν) consists of only one marker μ (resp. ν), delete the lone marker from the subsequences (recall that a strip must include at least two markers). This decreases the total strip length by at most two. Next insert the pair of dummy markers of index i to the subsequences as a new strip. This increases the total strip length by exactly two. Repeat this operation whenever a strip contains two vertex markers of different indices and whenever a pair of dummy markers is missing from the subsequences, then in $O(n)$ steps we obtain three subsequences of total strip length at least l in canonical form. □

The following lemma, analogous to Lemma 2, establishes the NP-hardness of MSR-3:

Lemma 5. *The graph G has an independent set of at least k vertices if and only if the three genomic maps G_0, G_1, G_2 have three subsequences whose total strip length l is at least $2(n+k)$.*

Proof. We first prove the "only if" direction. Suppose that the graph G has an independent set of at least k vertices. We will show that the three genomic maps G_0, G_1, G_2 have three subsequences of total strip length at least $2(n+k)$. By Proposition 1, the k vertices in the independent set correspond to k pairs of vertex markers that do not intersect each other in the genomic maps. These k pairs of vertex markers together with the n pairs of dummy markers induce a subsequence of length $2(n+k)$ in each genomic map. In each subsequence, the left marker and the right marker of each pair appear consecutively and compose a strip. Thus the total strip length is at least $2(n+k)$. We refer to Figure 2(c) for an example.

We next prove the "if" direction. Suppose that the three genomic maps G_0, G_1, G_2 have three subsequences of total strip length at least $2(n+k)$. We will show that the graph G has an independent set of at least k vertices. By Lemma 4, the three genomic maps have three subsequences of total strip length at least $2(n+k)$ such that each strip is a pair of markers. Excluding the n pairs of dummy markers, we obtain at least k pairs of vertex markers that do not intersect each other in the genomic maps. Then, by Proposition 1, the corresponding set of at least k vertices in the graph G form an independent set. □

4.2 L-reduction from Max-IS-3 to MSR-3

We present an L-reduction (f, g, α, β) from Max-IS-3 to MSR-3 as follows. The function f, given a graph G of maximum degree 3, constructs the three genomic maps G_0, G_1, G_2 as in the NP-hardness reduction. Let k^* be the number of vertices in a maximum independent set in G, and let l^* be the maximum total strip length of any three subsequences of G_0, G_1, G_2, respectively. Since a simple greedy algorithm (which repeatedly selects a vertex not adjacent to the previously selected vertices) finds an independent set of at least $n/(3+1)$ vertices in the graph G of maximum degree 3, we have $k^* \geq n/(3+1)$. By Lemma 5, we have $l^* = 2(n+k^*)$. It follows that

$$l^* = 2(n+k^*) \leq 2((3+1)k^* + k^*) = 2(3+2)k^* = 10k^*.$$

Choose $\alpha = 10$, then property (1) of L-reduction is satisfied.

The function g, given three subsequences of the three genomic maps G_0, G_1, G_2, respectively, transforms the subsequences into canonical form as in the proof of Lemma 4, then returns an independent set of vertices in the graph G corresponding to the pairs of vertex markers that are strips of the subsequences. Let l be the total strip length of the subsequences, and let k be the number of vertices in the independent set returned by the function g. Then $k \geq l/2 - n$. It follows that

$$|k^* - k| = k^* - k \leq (l^*/2 - n) - (l/2 - n) = |l^* - l|/2.$$

Choose $\beta = 1/2$, then property (2) of L-reduction is also satisfied.

We have obtained an L-reduction from Max-IS-3 to MSR-3 with $\alpha\beta = 5$. Chlebík and Chlebíková [12] showed that Max-IS-3 is NP-hard to approximate within $1.010661 = \frac{1}{1-(1-1/1.010661)}$. It follows that MSR-3 is NP-hard to approximate within $\frac{1}{1-(1-1/1.010661)/5} = 1.002114\ldots$.

5 MSR-2 is APX-hard

In this section, we prove that MSR-2 is APX-hard by an L-reduction from Ep-Occ-Max-Eq-SAT with $p = 3$ and $q \geq 2$.

5.1 NP-hardness reduction from Ep-Occ-Max-Eq-SAT to MSR-2

Let $(X, \mathcal{C})$ be an instance of Ep-Occ-Max-Eq-SAT, where X is a set of n variables x_i, $1 \le i \le n$, and $\mathcal{C}$ is a set of m clauses C_j, $1 \le j \le m$. Without loss of generality, assume that the p literals of each variable are neither all positive nor all negative. Since $p = 3$, it follows that each variable has either 2 positive and 1 negative literals, or 1 positive and 2 negative literals.

We construct two genomic maps G_1 and G_2, each map a permutation of $2(5n + m + qm + 2)$ distinct markers all in positive orientation:

- 1 pair of variable markers $\overset{i}{<}\ \overset{i}{>}$ for each variable x_i, $1 \le i \le n$;
- 2 pairs of true markers $\overset{i,1}{\blacktriangleleft}\ \overset{i,1}{\blacktriangleright}$ and $\overset{i,2}{\blacktriangleleft}\ \overset{i,2}{\blacktriangleright}$ for each variable x_i, $1 \le i \le n$;
- 2 pairs of false markers $\overset{i,1}{\triangleleft}\ \overset{i,1}{\triangleright}$ and $\overset{i,2}{\triangleleft}\ \overset{i,2}{\triangleright}$ for each variable x_i, $1 \le i \le n$;
- 1 pair of clause markers $\overset{j}{\Subset}\ \overset{j}{\Supset}$ for each clause C_j, $1 \le j \le m$;
- q pairs of literal markers $\overset{j,t}{\subset}\ \overset{j,t}{\supset}$, $1 \le t \le q$, for each clause C_j, $1 \le j \le m$;
- 2 pairs of dummy markers $\overset{1}{\sqsubset}\ \overset{1}{\sqsupset}$ and $\overset{2}{\sqsubset}\ \overset{2}{\sqsupset}$.

The construction is done in two steps: first arrange the variable markers, the true/false markers, the clause markers, and the dummy markers into two sequences $\check{G}_1$ and $\check{G}_2$, next insert the literal markers at appropriate positions in the two sequences to obtain the two genomic maps G_1 and G_2.

The two sequences $\check{G}_1$ and $\check{G}_2$ are represented schematically as follows:

$$\begin{array}{llllllllllllllllllll}
\check{G}_1: & \langle x_1 \rangle & \cdots & \langle x_n \rangle & \overset{1}{\sqsubset} & \overset{1}{\sqsupset} & \overset{2}{\sqsubset} & \overset{2}{\sqsupset} & \overset{1}{\Subset} & \overset{1}{\Supset} & \cdots & \overset{m}{\Subset} & \overset{m}{\Supset} & \overset{1}{<} & \overset{1}{>} & \cdots & \overset{n}{<} & \overset{n}{>} \\
\check{G}_2: & \langle x_n \rangle & \cdots & \langle x_1 \rangle & \overset{m}{\Subset} & \overset{m}{\Supset} & \cdots & \overset{1}{\Subset} & \overset{1}{\Supset} & \overset{2}{\sqsubset} & \overset{2}{\sqsupset} & \overset{1}{\sqsubset} & \overset{1}{\sqsupset}
\end{array}$$

For each variable x_i, $\langle x_i \rangle$ consists of the corresponding four pairs of true/false markers $\overset{i,1}{\blacktriangleleft}\ \overset{i,1}{\blacktriangleright}\ \overset{i,2}{\blacktriangleleft}\ \overset{i,2}{\blacktriangleright}\ \overset{i,1}{\triangleleft}\ \overset{i,1}{\triangleright}\ \overset{i,2}{\triangleleft}\ \overset{i,2}{\triangleright}$ in $\check{G}_1$ and $\check{G}_2$, and in addition the pair of variable markers $\overset{i}{<}\ \overset{i}{>}$ in $\check{G}_2$. These markers are arranged in the two sequences in a special pattern as follows (the indices i are omitted for simpler notations):

$$\begin{array}{cccccccccc}
 & \overset{1}{\triangleleft} & \overset{2}{\blacktriangleleft} & \overset{1}{\triangleright} & \overset{2}{\blacktriangleright} & \overset{2}{\triangleleft} & \overset{1}{\blacktriangleleft} & \overset{2}{\triangleright} & \overset{1}{\blacktriangleright} & \\
\overset{1}{\blacktriangleleft} & \overset{1}{\triangleleft} & \overset{1}{\blacktriangleright} & \overset{1}{\triangleright} & < \quad > & \overset{2}{\triangleleft} & \overset{2}{\blacktriangleleft} & \overset{2}{\triangleright} & \overset{2}{\blacktriangleright} &
\end{array}$$

Now insert the literal markers to the two sequences $\check{G}_1$ and $\check{G}_2$ to obtain the two genomic maps G_1 and G_2. First, $\check{G}_1 \to G_1$. For each positive literal (resp. negative literal) of a variable x_i that occurs in a clause C_j, place a pair of literal markers $\overset{j,t}{\subset}\ \overset{j,t}{\supset}$, $1 \le t \le q$, around a false marker $\overset{i,s}{\triangleleft}$ (resp. true marker $\overset{i,s}{\blacktriangleright}$), $1 \le s \le 2$. The four possible positions of the three pairs of literal markers of each variable x_i are as follows:

$$\begin{array}{cccccccccc}
 & \subset\overset{1}{\triangleleft}\supset & \overset{2}{\blacktriangleleft} & \overset{1}{\triangleright} & \subset\overset{2}{\blacktriangleright}\supset & \subset\overset{2}{\triangleleft}\supset & \overset{1}{\blacktriangleleft} & \overset{2}{\triangleright} & \subset\overset{1}{\blacktriangleright}\supset & \\
\overset{1}{\blacktriangleleft} & \overset{1}{\triangleleft} & \overset{1}{\blacktriangleright} & \overset{1}{\triangleright} & < \quad > & \overset{2}{\triangleleft} & \overset{2}{\blacktriangleleft} & \overset{2}{\triangleright} & \overset{2}{\blacktriangleright} &
\end{array}$$

Next, $\check{G}_2 \to G_2$. Without loss of generality, assume that the q pairs of literal markers of each clause C_j appear in G_1 with ascending indices:

$$\overset{j,1}{\subset}\quad \overset{j,1}{\supset}\quad \cdots \quad \overset{j,q}{\subset}\quad \overset{j,q}{\supset}$$

Insert the q pairs of literal markers in G_2 immediately after the pair of clause markers $\overset{j}{\Subset}\ \overset{j}{\Supset}$, in an interleaving pattern:

$$\overset{j,q}{\subset}\quad \cdots \quad \overset{j,1}{\subset}\quad \overset{j,q}{\supset}\quad \cdots \quad \overset{j,1}{\supset}$$

This completes the construction. We refer to Figure 3 (a) and (b) for an example of the two steps.

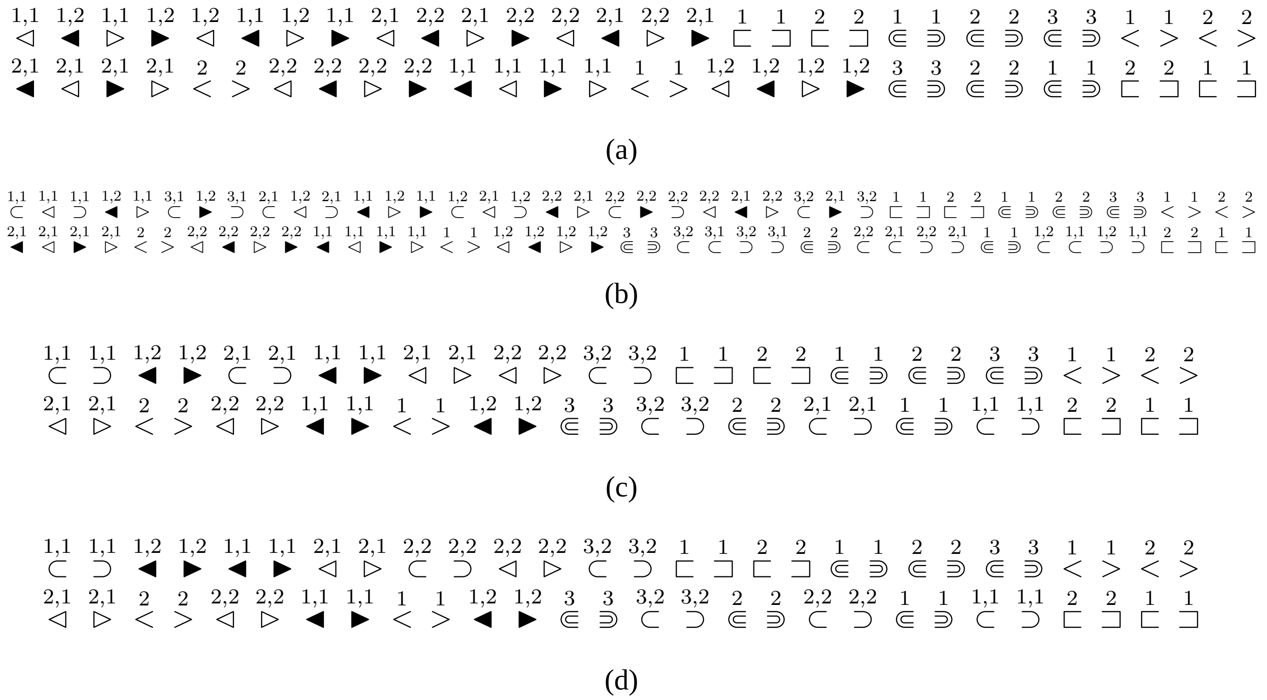

Figure 3: MSR-2 construction for the E3-Occ-Max-E2-SAT instance $C_1 = x_1 \vee x_2$, $C_2 = x_1 \vee \bar{x}_2$, and $C_3 = \bar{x}_1 \vee \bar{x}_2$. (a) The two sequences $\check{G}_1$ and $\check{G}_2$. (b) The two genomic maps G_1 and G_2. (c) Two canonical subsequences for the assignment $x_1 = \mathit{true}$ and $x_2 = \mathit{false}$. (d) Two other canonical subsequences for the assignment $x_1 = \mathit{true}$ and $x_2 = \mathit{false}$.

We say that two subsequences of the two genomic maps G_1 and G_2 are *canonical* if each strip of the two subsequences is a pair of markers. We refer to Figure 3 (c) and (d) for two examples of canonical subsequences. The following lemma on canonical subsequences is analogous to Lemma 1 and Lemma 4:

Lemma 6. *If the two genomic maps G_1 and G_2 have two subsequences of total strip length l, then they must have two subsequences of total strip length at least l such that each strip is a pair of markers and, moreover,* (i) *the two pairs of dummy markers are two strips,* (ii) *the m pairs of clause markers and the n pairs of variable markers are $m + n$ strips,* (iii) *at most one pair of literal markers of each clause is a strip,* (iv) *either both pairs of true markers or both pairs of false markers of each variable are two strips.*

Proof. We present an algorithm that transforms the subsequences into canonical form without reducing the total strip length. The algorithm performs incremental operations on the subsequences such that the following eight conditions are satisfied progressively:

1. Each strip that includes a dummy marker is a pair of dummy markers. A strip cannot include two dummy markers of different indices because they appear in different orders in G_1 and in G_2. Note that in G_2 the dummy markers appear after the other markers. Suppose that a strip S includes both a dummy marker and a non-dummy marker. Then there must be a non-dummy marker μ and a dummy marker ν consecutive in S. Since the two pairs of dummy markers appear consecutively but in different orders in G_1 and in G_2, one of the two pairs must appear between μ and ν either in G_1 or in G_2. This pair is hence missing from the subsequences. Now cut the strip S into S_μ and S_ν between μ and ν. If S_μ (resp. S_ν) consists of only one marker μ (resp. ν), delete the lone marker from the subsequences (recall that a strip must include at least two markers). This decreases the total strip length by at most two. Next insert the missing pair of dummy markers to the subsequences. This pair of dummy markers becomes either a new strip by itself, or part of a longer strip (recall that a strip must be maximal). In any case, the insertion increases the total strip length by exactly two. Overall, this *cut-delete-insert* operation (also used in Lemma 4) does not reduce the total strip length. After the first operation, a second operation may be necessary. But since each operation here deletes only lone markers (in S_μ and S_ν) and inserts always a pair of markers, the pair inserted by one operation is never deleted by a subsequent operation. Thus at most two operations are sufficient to transform the subsequences until each strip that includes a dummy marker is indeed a pair of dummy markers.

2. The two pairs of dummy markers are two strips. Suppose that the subsequences do not have both pairs of dummy markers as strips. Then, by condition 1, we must have either both pairs of dummy markers missing from the subsequences, or one pair missing and the other pair forming a strip. Note that in G_1 the dummy markers separate the true/false and literal markers on the left from the clause and variable markers on the right, and that in G_2 the dummy markers appear after the other markers. If the missing dummy markers do not disrupt any existing strips in G_1, then simply insert each missing pair to the subsequences as a new strip. Otherwise, there must be a true/false or literal marker μ and a clause or variable marker ν consecutive in a strip S, such that both pairs of dummy markers appear in G_1 between μ and ν and hence are missing from the subsequences. Cut the strip S between μ and ν, delete any lone markers if necessary, then insert the two pairs of dummy markers to the subsequences as two new strips.

3. Each strip that includes a clause or variable marker is a pair of clause markers or a pair of variable markers. Note that in G_1 the clause and variable markers are separated by the dummy markers from the other markers. Thus, by condition 2, a strip that includes a clause or variable marker cannot include any markers of the other types. Also, a strip cannot include two clause markers of different clauses, or two variable markers of different variables, or a clause marker and a variable marker, because these combinations appear in different orders in G_1 and in G_2. Thus this condition is automatically satisfied after conditions 1 and 2.

4. The m pairs of clause markers and the n pairs of variable markers are $m+n$ strips. Suppose that the subsequences do not have all $m+n$ pairs of clause and variable markers as $m+n$ strips. By condition 3, the clause and variable markers in the subsequences must be in pairs, each pair forming a strip. Then the clause and variable markers missing from the subsequences must be in pairs too. For each missing pair of clause or variable markers, if the pair does not disrupt any existing strips in G_2, then simply insert it to the subsequences as a new strip. Otherwise, there must be two true/false or literal markers μ and ν consecutive in a strip S, such that the missing pair appears in G_2 between μ and ν. Cut the strip S between μ and ν, delete any lone markers if necessary, then insert each missing pair of clause markers between μ and ν to the subsequences as a new strip.

5. Each strip that includes a literal marker is a pair of literal markers. Note that in G_2 the dummy and clause markers separate the literals markers from the other markers, and separate the literal markers of different clauses from each other. Thus, by conditions 2 and 4, a strip cannot include both a literal marker and a non-literal marker, or two literal markers of different clauses. Suppose that a strip S includes two literal markers μ and ν of the same clause C_j but of different indices j,s and j,t. Assume without loss of generality that μ and ν are consecutive in S. Recall the orders of the literal markers of each clause in the two genomic maps:

$$\overset{j,1}{\subset}\ \overset{j,1}{\supset}\ \cdots\ \underline{\overset{j,s}{\subset}\ \overset{j,s}{\supset}\ \cdots\ \overset{j,t}{\subset}\ \overset{j,t}{\supset}}\ \cdots\ \overset{j,q}{\subset}\ \overset{j,q}{\supset}$$
$$\overset{j,q}{\subset}\ \cdots\ \underline{\overset{j,t}{\subset}\ \cdots\ \overset{j,s}{\subset}}\ \cdots\ \overset{j,1}{\subset}\ \overset{j,q}{\supset}\ \cdots\ \underline{\overset{j,t}{\supset}\ \cdots\ \overset{j,s}{\supset}}\ \cdots\ \overset{j,1}{\supset}$$

Since in G_1 the pairs of literal markers appear with ascending indices, the index s of the marker μ must be less than the index t of the marker ν. Then, since in G_2 the left markers appear with descending indices before the right markers also with descending indices, μ must be a left marker, and ν must be a right marker. That is, $\mu\nu = \overset{j,s}{\subset}\,\overset{j,t}{\supset}$. All markers between μ and ν in G_1 must be missing from the subsequences. Among these missing markers, those that are literal markers of C_j appear in G_2 either consecutively before μ or consecutively after ν. Replace either μ or ν by a missing literal marker of C_j, that is, either $\overset{j,s}{\subset}$ by $\overset{j,t}{\subset}$, or $\overset{j,t}{\supset}$ by $\overset{j,s}{\supset}$, then μ and ν become a pair. Denote this *shift* operation by

$$\mu\nu : \quad \overset{j,s}{\subset}\,\overset{j,t}{\supset} \to \overset{j,t}{\subset}\,\overset{j,t}{\supset} \text{ or } \overset{j,s}{\subset}\,\overset{j,s}{\supset}\ .$$

The strip S cannot include any other literal markers of the clause C_j besides μ and ν because (i) the markers before $\overset{j,s}{\subset}$ in G_1 appear after $\overset{j,s}{\subset}$ in G_2, and (ii) the markers after $\overset{j,t}{\supset}$ in G_1 appear before $\overset{j,t}{\supset}$ in G_2.

6. At most one pair of literal markers of each clause is a strip. Note that the q pairs of literal markers of each clause appear in G_2 in an interleaving pattern. It follows by condition 5 that at most one of the q pairs can be a strip.

7. Each strip that includes a true/false marker is a pair of true markers or a pair of false markers. By conditions 1, 3, and 5, it follows that each strip that includes a true/false marker must include true/false markers only. A strip cannot include two true/false markers of different variables because they appear in different orders in G_1 and in G_2. Suppose that a strip S includes two true/false markers μ and ν of the same variable x_i such that μ and ν are not a pair. Recall the orders of the four pairs of true/false markers of each variable x_i in G_1 and G_2, the four possible positions of the three pairs of literal markers in G_1, and the position of the variable marker in G_2:

$$\begin{array}{c}
\subset\overset{1}{\lhd}\supset \quad \overset{2}{\blacktriangleleft} \quad \overset{1}{\rhd} \quad \subset\overset{2}{\blacktriangleright}\supset \qquad \subset\overset{2}{\lhd}\supset \quad \overset{1}{\blacktriangleleft} \quad \overset{2}{\rhd} \quad \subset\overset{1}{\blacktriangleright}\supset \\
\overset{1}{\blacktriangleleft} \quad \overset{1}{\lhd} \quad \overset{1}{\blacktriangleright} \quad \overset{1}{\rhd} \quad < \quad > \quad \overset{2}{\lhd} \quad \overset{2}{\blacktriangleleft} \quad \overset{2}{\rhd} \quad \overset{2}{\blacktriangleright}
\end{array}$$

Note that the pair of variable markers in G_2 forbids a strip from including two true/false markers of different indices. Thus the strip S must consist of true/false markers of both the same variable and the same index. Assume without loss of generality that μ appears before ν in S. It is easy to check that there are only two such combinations of μ and ν: either $\mu\nu = \overset{1}{\lhd}\,\overset{1}{\blacktriangleright}$ or $\mu\nu = \overset{2}{\blacktriangleleft}\,\overset{2}{\rhd}$. Moreover, the strip S must include only the two markers μ and ν. For either combination of μ and ν, use a shift operation to make μ and ν a pair:

$$\begin{array}{rl}
\mu\nu : & \overset{1}{\lhd}\,\overset{1}{\blacktriangleright} \to \overset{1}{\lhd}\,\overset{1}{\rhd} \text{ or } \overset{1}{\blacktriangleleft}\,\overset{1}{\blacktriangleright} \\
\mu\nu : & \overset{2}{\blacktriangleleft}\,\overset{2}{\rhd} \to \overset{2}{\blacktriangleleft}\,\overset{2}{\blacktriangleright} \text{ or } \overset{2}{\lhd}\,\overset{2}{\rhd}\ .
\end{array}$$

8. Either both pairs of true markers or both pairs of false markers of each variable are two strips. Consider the conflict graph of the four pairs of true/false markers and the three pairs of literal markers of each variable x_i in Figure 4. The graph has one vertex for each pair, and has an edge between two vertices if and only if the corresponding pairs intersect in either G_1 or G_2. By conditions 1, 3, 5, and 7, the strips of the subsequences from the seven pairs correspond to an independent set in the conflict graph of seven vertices.

Note that the four vertices corresponding to the four pairs of true/false markers induce a 4-cycle in the conflict graph. Suppose that neither both pairs of true markers nor both pairs of false markers are strips. Then at most one of the four pairs, say S, is a strip. Delete S from the subsequences. Recall that each variable has either 2 positive and 1 negative literals, or 1 positive and 2 negative literals. Let T be the pair of literal markers whose sign is opposite to the sign of the other two pairs of literal markers. Also delete T from the subsequences if it is there. Next insert two pairs of true/false markers to the subsequences: if T is

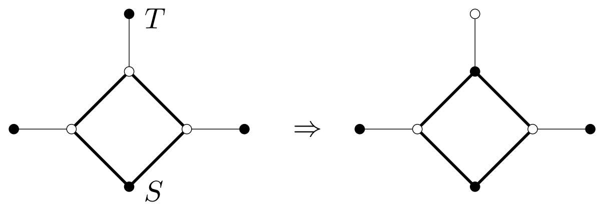

Figure 4: Replacing vertices of the independent set in the conflict graph of the four pairs of true/false markers and the three pairs of literal markers of each variable. Vertices in the independent set are black. Edges in the 4-cycle are thick. In this example the strip S is first deleted then inserted back.

positive, both pairs of false markers $\overset{i,1}{\lhd}\ \overset{i,1}{\rhd}$ and $\overset{i,2}{\lhd}\ \overset{i,2}{\rhd}$; if T is negative, both pairs of true markers $\overset{i,1}{\blacktriangleleft}\ \overset{i,1}{\blacktriangleright}$ and $\overset{i,2}{\blacktriangleleft}\ \overset{i,2}{\blacktriangleright}$.

When all eight conditions are satisfied, the subsequences are in the desired canonical form. □

The following lemma, analogous to Lemma 2 and Lemma 5, establishes the NP-hardness of MSR-2:

Lemma 7. *The variables in X have an assignment that satisfies at least k clauses in $\mathcal{C}$ if and only if the two genomic maps G_1 and G_2 have two subsequences whose total strip length l is at least $2(3n + m + k + 2)$.*

Proof. We first prove the "only if" direction. Suppose that the variables in X have an assignment that satisfies at least k clauses in $\mathcal{C}$. We will show that the two genomic maps G_1 and G_2 have two subsequences of total strip length at least $2(3n + m + k + 2)$. For each variable x_i, choose the two pairs of true markers if the variable is assigned true, or the two pairs of false markers if the variable is assigned false. For each satisfied clause C_j, choose one pair of literal markers corresponding to a true literal (when there are two or more true literals, choose any one). Also choose all $m + n$ pairs of clause and variable markers and both pairs of dummy markers. The chosen markers induce two subsequences of the two genomic maps. It is easy to check that, by construction, the two subsequences have at least $3n + m + k + 2$ strips, each strip forming a pair. Thus the total strip length is at least $2(3n + m + k + 2)$. We refer to Figure 3 (c) and (d) for two examples.

We next prove the "if" direction. Suppose that the two genomic maps G_1 and G_2 have two subsequences of total strip length at least $2(3n+m+k+2)$. We will show that the variables in X have an assignment that satisfies at least k clauses in $\mathcal{C}$. By Lemma 6, the two genomic maps have two subsequences of total strip length at least $2(3n+m+k+2)$ such that each strip is a pair and, moreover, the two pairs of dummy markers, the $m + n$ pairs of clause and variable markers, at most one pair of literal markers of each clause, and either both pairs of true markers or both pairs of false markers of each variable are strips. Thus at least k strips are pairs of literal markers, each pair of a different clause. Again it is easy to check that, by construction, the assignment of the variables in X to either true or false (corresponding to the choices of either both pairs of true markers or both pairs of false markers) satisfies at least k clauses in $\mathcal{C}$ (corresponding to the at least k pairs of literal markers that are strips). □

5.2 L-reduction from Ep-Occ-Max-Eq-SAT to MSR-2

We present an L-reduction (f, g, α, β) from Ep-Occ-Max-Eq-SAT to MSR-3 as follows. The function f, given the Ep-Occ-Max-Eq-SAT instance $(X, \mathcal{C})$, constructs the two genomic maps G_1 and G_2 as in the NP-hardness reduction. Let k^* be the maximum number of clauses in $\mathcal{C}$ that can be satisfied by an assignment of X, and let l^* be the maximum total strip length of any two subsequences of G_1 and G_2, respectively. Since a random assignment of each variable independently to either true or false with equal probability $\frac{1}{2}$ satisfies each disjunctive clause of q literals with probability $1 - \frac{1}{2^q}$, we have $k^* \geq \frac{2^q-1}{2^q} m$. By Lemma 7, we have $l^* = 2(3n + m + k^* + 2)$. Recall that $np = mq$. It follows that

$$l^* = 2(3n + m + k^* + 2) = \left(6\,\frac{q}{p} + 2\right) m + 2k^* + 4 \leq \left(\left(6\,\frac{q}{p} + 2\right) \frac{2^q}{2^q - 1} + 2 + \frac{4}{k^*}\right) k^*.$$

The function g, given two subsequences of the two genomic maps G_1 and G_2, respectively, transforms the subsequences into canonical form as in the proof of Lemma 6, then returns an assignment of X corresponding to the choices of true or false markers. Let l be the total strip length of the subsequences, and let k be the number of clauses in $\mathcal{C}$ that are satisfied by this assignment. Then $k \geq l/2 - 3n - m - 2$. It follows that

$$|k^* - k| = k^* - k \leq (l^*/2 - 3n - m - 2) - (l/2 - 3n - m - 2) = |l^* - l|/2.$$

Let $\epsilon > 0$ be an arbitrary small constant. Note that by brute force we can check whether $k^* < 2/\epsilon$ and, in the affirmative case, compute an optimal assignment of X that satisfies the maximum number of clauses in $\mathcal{C}$, all in $m^{O(1/\epsilon)}$ time, which is polynomial in m for a constant ϵ. Therefore we can assume without loss of generality that $k^* \geq 2/\epsilon$. Then, with the two constants $\alpha = (6\,\frac{q}{p} + 2)\frac{2^q}{2^q-1} + 2 + 2\epsilon$ and $\beta = 1/2$, both properties (1) and (2) of L-reduction are satisfied. In particular, for $p = 3$ and $q = 2$,

$$\alpha\beta = \left(3\frac{q}{p} + 1\right) \frac{2^q}{2^q - 1} + 1 + \epsilon = 5 + \epsilon.$$

Berman and Karpinski [8] showed that E3-Occ-Max-E2-SAT is NP-hard to approximate within any constant less than $\frac{464}{463} = \frac{1}{1-1/464}$. Thus MSR-2 is NP-hard to approximate within any constant less than

$$\lim_{\epsilon \to 0} \frac{1}{1 - (1/464)/(5+\epsilon)} = \frac{1}{1 - 1/2320} = \frac{2320}{2319} = 1.000431\ldots.$$

6 An asymptotic lower bound for MSR-d

In this section, we derive an asymptotic lower bound for approximating MSR-d by an L-reduction from d-Dimensional-Matching to MSR-$(d+2)$.

6.1 NP-hardness reduction from d-Dimensional-Matching to MSR-$(d+2)$

Let $E \subseteq V_1 \times \cdots \times V_d$ be a set of n hyper-edges over d disjoint sets V_i of vertices, $1 \leq i \leq d$. We construct two genomic maps $G_\rightarrow$ and $G_\leftarrow$, and d genomic maps G_i, $1 \leq i \leq d$, where each map is a permutation of the following $2n$ distinct markers all in positive orientation:

- n pairs of edge markers $\overset{i}{\subset}$ and $\overset{i}{\supset}$, $1 \leq i \leq n$.

The two genomic maps $G_\rightarrow$ and $G_\leftarrow$ are concatenations of the n pairs of edge markers with ascending and descending indices, respectively:

$$\begin{array}{llll} G_\rightarrow : & \overset{1}{\subset}\ \overset{1}{\supset} & \cdots & \overset{n}{\subset}\ \overset{n}{\supset} \\ G_\leftarrow : & \overset{n}{\subset}\ \overset{n}{\supset} & \cdots & \overset{1}{\subset}\ \overset{1}{\supset} \end{array}$$

Each genomic map G_i corresponds to a vertex set $V_i = \{v_{i,j} \mid 1 \leq j \leq |V_i|\}$, $1 \leq i \leq d$, and is represented schematically as follows:

$$G_i : \quad \cdots \quad \langle v_{i,j} \rangle \quad \cdots$$

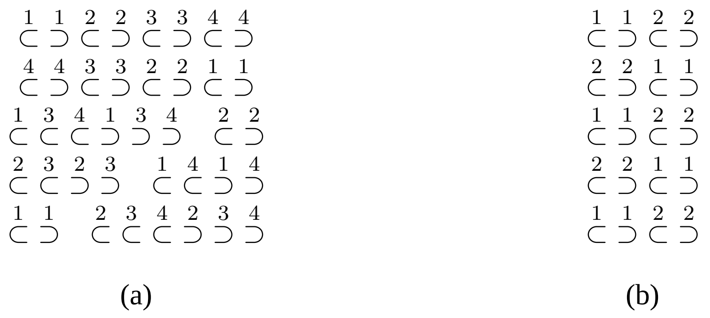

Figure 5: MSR-5 construction for the 3-Dimensional-Matching instance $V_1 = \{v_{1,1}, v_{1,2}\}$, $V_2 = \{v_{2,1}, v_{2,2}\}$, $V_3 = \{v_{3,1}, v_{3,2}\}$, and $E = \{\, e_1 = (v_{1,1}, v_{2,2}, v_{3,1}),\ e_2 = (v_{1,2}, v_{2,1}, v_{3,2}),\ e_3 = (v_{1,1}, v_{2,1}, v_{3,2}),\ e_4 = (v_{1,1}, v_{2,2}, v_{3,2}) \,\}$. (a) The five genomic maps $G_\to, G_\leftarrow, G_1, G_2, G_3$. (b) The five subsequences of the genomic maps corresponding to the subset $\{e_1, e_2\}$ of pairwise-disjoint hyper-edges.

Here each $\langle v_{i,j} \rangle$ consists of the edge markers of hyper-edges containing the vertex $v_{i,j}$, grouped together such that the left markers appear with ascending indices before the right markers also with ascending indices. This completes the construction. We refer to Figure 5(a) for an example.

The following property of our construction is obvious:

Proposition 2. *Two hyper-edges in E intersect if and only if the corresponding two pairs of edge markers intersect in one of the d genomic maps G_i, $1 \le i \le d$.*

The following lemma is analogous to Lemma 1:

Lemma 8. *In any $d + 2$ subsequences of the $d + 2$ genomic maps $G_\to, G_\leftarrow, G_1, \ldots, G_d$, respectively, each strip must be a pair of edge markers.*

Proof. By construction, a strip cannot include two edge markers of different indices because they appear in different orders in $G_\to$ and in $G_\leftarrow$. □

The following lemma, analogous to Lemma 2, Lemma 5, and Lemma 7, establishes the NP-hardness of MSR-d:

Lemma 9. *The set E has a subset of k pairwise-disjoint hyper-edges if and only if the $d + 2$ genomic maps $G_\to, G_\leftarrow, G_1, \ldots, G_d$ have $d + 2$ subsequences whose total strip length l is at least $2k$.*

Proof. We first prove the "only if" direction. Suppose that the set E has a subset of at least k pairwise-disjoint hyper-edges. We will show that the $d + 2$ genomic maps $G_\to, G_\leftarrow, G_1, \ldots, G_d$ have $d + 2$ subsequences of total strip length at least $2k$. By Proposition 2, the k pairwise-disjoint hyper-edges correspond to k pairs of edge markers that do not intersect each other in the genomic maps. These k pairs of edge markers induce a subsequence of length $2k$ in each genomic map. In each subsequence, the left marker and the right marker of each pair appear consecutively and compose a strip. Thus the total strip length is at least $2k$. We refer to Figure 5(b) for an example.

We next prove the "if" direction. Suppose that the $d + 2$ genomic maps $G_\to, G_\leftarrow, G_1, \ldots, G_d$ have $d + 2$ subsequences of total strip length at least $2k$. We will show that the set E has a subset of at least k pairwise-disjoint hyper-edges. By Lemma 1, each strip of the subsequences must be a pair of edge markers. Thus we obtain at least k pairs of edge markers that do not intersect each other in the genomic maps. Then, by Proposition 2, the corresponding set of at least k hyper-edges in E are pairwise-disjoint. □

6.2 L-reduction from d-Dimensional-Matching to MSR-$(d+2)$

We present an L-reduction (f, g, α, β) from d-Dimensional-Matching to MSR-$(d+2)$ as follows. The function f, given a set $E \subseteq V_1 \times \cdots \times V_d$ of hyper-edges, constructs the $d+2$ genomic maps $G_\rightarrow, G_\leftarrow, G_1, \ldots, G_d$ as in the NP-hardness reduction. Let k^* be the maximum number of pairwise-disjoint hyper-edges in E, and let l^* be the maximum total strip length of any $d+2$ subsequences of $G_\rightarrow, G_\leftarrow, G_1, \ldots, G_d$, respectively. By Lemma 9, we have

$$l^* = 2k^*.$$

Choose $\alpha = 2$, then property (1) of L-reduction is satisfied.

The function g, given $d+2$ subsequences of the $d+2$ genomic maps $G_\rightarrow, G_\leftarrow, G_1, \ldots, G_d$, respectively, returns a subset of pairwise-disjoint hyper-edges in E corresponding to the pairs of edge markers that are strips of the subsequences. Let l be the total strip length of the subsequences, and let k be the number of pairwise-disjoint hyper-edges returned by the function g. Then $k \geq l/2$. It follows that

$$|k^* - k| = k^* - k \leq l^*/2 - l/2 = |l^* - l|/2.$$

Choose $\beta = 1/2$, then property (2) of L-reduction is also satisfied.

We have obtained an L-reduction from d-Dimensional-Matching to MSR-$(d+2)$ with $\alpha\beta = 1$. Hazan, Safra, and Schwartz [16] showed that d-Dimensional-Matching is NP-hard to approximate within $\Omega(d/\log d)$. It follows that MSR-d is also NP-hard to approximate within $\Omega(d/\log d)$. This completes the proof of Theorem 1.

7 A polynomial-time $2d$-approximation for MSR-d

In this section we prove Theorem 2. We briefly review the two previous algorithms [27, 11] for this problem. The first algorithm for MSR-2 is a simple heuristic due to Zheng, Zhu, and Sankoff [27]:

1. Extract a set of pre-strips from the two genomic maps;
2. Compute an independent set of strips from the pre-strips.

This algorithm is inefficient because the number of pre-strips could be exponential in the sequence length, and furthermore the problem Maximum-Weight Independent Set in general graphs is NP-hard.

Chen, Fu, Jiang, and Zhu [11] presented a $2d$-approximation algorithm for MSR-d. For any $d \geq 2$, a d-*interval* is the union of d disjoint intervals in the real line, and a d-*interval graph* is the intersection graph of a set of d-intervals, with a vertex for each d-interval, and with an edge between two vertices if and only the corresponding d-intervals overlap. The $2d$-approximation algorithm [11] works as follows:

1. Compose a set of d-intervals, one for each combination of d substrings of the d genomic maps, respectively. Assign each d-interval a weight equal to the length of a longest common subsequence (which may be reversed and negated) in the corresponding d substrings.
2. Compute a $2d$-approximation for Maximum-Weight Independent Set in the resulting d-interval graph using Bar-Yehuda et al.'s fractional local-ratio algorithm [6].

Let n be the number of markers in each genomic map. Then the number of d-intervals composed by this algorithm is $\Theta(n^{2d})$ because each of the d genomic maps has $\Theta(n^2)$ substrings. Consequently the running time of this algorithm can be exponential if the number d of genomic maps is not a constant but is part of the input. In the following, we show that if all markers are distinct in each genomic map (as discussed earlier, this is a reasonable assumption in application), then the running time of the $2d$-approximation algorithm can be improved to polynomial for all $d \geq 2$. This improvement is achieved by composing a smaller set of candidate d-intervals in step 1 of the algorithm.

The idea is actually quite simple and has been used many times previously [21, 19, 10]. Note that any strip of length $l > 3$ is a concatenation of shorter strips of lengths 2 and 3, for example, $4 = 2 + 2$, $5 = 2 + 3$, etc. Since the objective is to maximize the total strip length, it suffices to consider only short strips of lengths 2 and 3 in the genomic maps, and to enumerate only candidate d-intervals that correspond to these strips. When each genomic map is a signed permutation of the same n distinct markers, there are at most $\binom{n}{2} + \binom{n}{3} = O(n^3)$ strips of lengths 2 and 3, and for each strip there is a unique shortest substring of each genomic map that contains all markers in the strip. Thus we compose only $O(n^3)$ d-intervals, and improve the running time of the $2d$-approximation algorithm to polynomial for all $d \geq 2$. This completes the proof of Theorem 2.

8 Inapproximability results for related problems

In this section we prove Theorem 3 and Theorem 4.

CMSR-3 and CMSR-4 are APX-hard. For any d, the decision problems of MSR-d and CMSR-d are equivalent. Thus the NP-hardness of MSR-d implies the NP-hardness of CMSR-d, although the APX-hardness of MSR-d does not necessarily imply the APX-hardness of CMSR-d. Note that the two problems Max-IS-Δ and Min-VC-Δ complement each other just as the two problems MSR-d and CMSR-d complement each other. Thus our NP-hardness reduction from Max-IS-3 to MSR-3 in Section 4 can be immediately turned into an NP-hardness reduction from Min-VC-3 to CMSR-3.

We present an L-reduction (f, g, α, β) from Min-VC-3 to CMSR-3 as follows. The function f, given a graph G of maximum degree 3, constructs the three genomic maps G_0, G_1, G_2 as in the NP-hardness reduction in Section 4. Let k^* be the number of vertices in a maximum independent set in G, and let l^* be the maximum total strip length of any three subsequences of G_0, G_1, G_2, respectively. Also let c^* be the number of vertices in a minimum vertex cover in G, and let x^* be the minimum number of markers that must be deleted to transform the three genomic maps G_0, G_1, G_2 into strip-concatenated subsequences. Then $k^* + c^* = n$ and $l^* + x^* = 4n$. By Lemma 5, we have $l^* = 2(n + k^*)$. It follows that

$$x^* = 4n - l^* = 4n - 2(n + k^*) = 2(n - k^*) = 2c^*.$$

Choose $\alpha = 2$, then property (1) of L-reduction is satisfied.

The function g, given three subsequences of the three genomic maps G_0, G_1, G_2, respectively, transforms the subsequences into canonical form as in the proof of Lemma 4, then returns a vertex cover in the graph G corresponding to the deleted pairs of vertex markers. Let x be the number of deleted vertex markers, and let c be the number of vertices in the vertex cover returned by the function g. Then $c \leq x/2$. It follows that

$$|c^* - c| = c - c^* \leq x/2 - x^*/2 = |x^* - x|/2.$$

Choose $\beta = 1/2$, then property (2) of L-reduction is also satisfied.

The L-reduction from Min-VC-3 to CMSR-3 can be obviously generalized:

Lemma 10. *Let $\Delta \geq 3$ and $d \geq 3$. If there is a polynomial-time algorithm for decomposing any graph of maximum degree Δ into $d - 1$ linear forests, then there is an L-reduction from* Min-VC-Δ *to* CMSR-d *with constants $\alpha = 2$ and $\beta = 1/2$.*

Recall that there exist polynomial-time algorithms for decomposing a graph of maximum degree 3 and 4 into at most 2 and 3 linear forests, respectively [2, 1, 3]. Thus we have an L-reduction from Min-VC-3 to CMSR-3 and an L-reduction from Min-VC-4 to CMSR-4, with the same parameters $\alpha = 2$, $\beta =$

$1/2$, and $\alpha\beta = 1$. Chlebík and Chlebíková [12] showed that Min-VC-3 and Min-VC-4 are NP-hard to approximate within 1.0101215 and 1.0202429, respectively. It follows that CMSR-3 and CMSR-4 are NP-hard to approximate within 1.0101215 and 1.0202429, respectively, too. The lower bound for CMSR-4 extends to CMSR-d for all $d \geq 4$. Note that we could use an L-reduction from Min-VC-3 to CMSR-4 similar to the L-reduction from Max-IS-3 to MSR-4 in Section 3, but that only gives us a weaker lower bound of 1.0101215 for CMSR-4.

CMSR-2 is APX-hard. Let $p = 3$ and $q \geq 2$. We present an L-reduction (f, g, α, β) from Ep-Occ-Max-Eq-SAT to CMSR-2 as follows. The function f, given the Ep-Occ-Max-Eq-SAT instance $(X, \mathcal{C})$, constructs the two genomic maps G_1 and G_2 as in our NP-hardness reduction in Section 5. As before, let k^* be the maximum number of clauses in $\mathcal{C}$ that can be satisfied by an assignment of X, and let l^* be the maximum total strip length of any two subsequences of G_1 and G_2, respectively. Also let x^* be the minimum number of deleted markers. Then $l^* + x^*$ is exactly the number of markers in each genomic map, that is, $2(5n + m + qm + 2)$. By Lemma 7, we have $l^* = 2(3n + m + k^* + 2)$. Thus $x^* = 2(5n + m + qm + 2) - 2(3n + m + k^* + 2) = 2(2n + qm - k^*)$. Since a random assignment of each variable independently to either true or false with equal probability $\frac{1}{2}$ satisfies each disjunctive clause of q literals with probability $1 - \frac{1}{2^q}$, we have $k^* \geq \frac{2^q-1}{2^q}m$. Recall that $np = mq$. It follows that

$$x^* = 2(2n + qm - k^*) = 2\left(2\,\frac{q}{p} + q\right)m - 2k^* \leq \left(2\left(2\,\frac{q}{p} + q\right)\frac{2^q}{2^q - 1} - 2\right)k^*.$$

For $p = 3$ and $q = 2$, we can choose $\alpha = 2(2\,\frac{q}{p} + q)\frac{2^q}{2^q-1} - 2 = 62/9$. Then property (1) of L-reduction is satisfied.

The function g, given two subsequences of the two genomic maps G_1 and G_2, transforms the subsequences into canonical form as in the proof of Lemma 6, then returns an assignment of X corresponding to the choices of true or false markers. Let l be the total strip length of the subsequences, and let x be the number of deleted markers. Let k be the number of clauses in $\mathcal{C}$ that are satisfied by this assignment. Then

$$|k^* - k| \leq |l^* - l|/2 = |x^* - x|/2.$$

Choose $\beta = 1/2$. then property (2) of L-reduction is satisfied.

Berman and Karpinski [8] showed that E3-Occ-Max-E2-SAT is NP-hard to approximate within any constant less than $\frac{464}{463} = \frac{1}{1-1/464}$. Since $\alpha\beta = 31/9$, CMSR-2 is NP-hard to approximate within any constant less than

$$1 + (1/464)/(31/9) = 1 + 9/14384 = 1.000625\ldots.$$

An asymptotic lower bound for CMSR-d and a lower bound for CMSR-d with unbounded d. Chlebík and Chlebíková [12] showed that for any $\Delta \geq 228$, Min-VC-Δ is NP-hard to approximate within $\frac{7}{6} - O(\log\Delta/\Delta)$. By the second inequality in (3), it follows that if $\Delta \leq 227$, then $f(\Delta) \leq \lceil 3\lceil 227/2\rceil/2\rceil = 171$. Consequently, if $f(\Delta) \geq 172$, then $\Delta \geq 228$. By Lemma 10, there is an L-reduction from Min-VC-Δ to CMSR-$(f(\Delta) + 1)$ with $\alpha = 2$ and $\beta = 1/2$. Therefore, for any $d \geq 173$, CMSR-d is NP-hard to approximate within $\frac{7}{6} - O(\log d/d)$.

The maximum degree Δ of a graph of n vertices is at most $n - 1$. Again by the second inequality in (3), we have $f(\Delta) \leq \lceil 3\lceil (n-1)/2\rceil/2\rceil$. Thus $f(\Delta)$ is bounded by a polynomial in n. If d is not a constant but is part of the input, then a straightforward generalization of the L-reduction from Min-VC-3 to CMSR-3 as in Lemma 10 gives an L-reduction from Minimum Vertex Cover to CMSR-$(f(\Delta) + 1)$ with $\alpha = 2$ and $\beta = 1/2$. Dinur and Safra [14] showed that Minimum Vertex Cover is NP-hard to approximate within any constant less than $10\sqrt{5} - 21 = 1.3606\ldots$. It follows that if d is not a constant but is part of the input, then CMSR-d is NP-hard to approximate within any constant less than $10\sqrt{5} - 21 = 1.3606\ldots$. This completes the proof of Theorem 3.

Inapproximability of δ-gap-MSR-d and δ-gap-CMSR-d. It is easy to check that all instances of MSR-d and CMSR-d in our constructions for Theorem 1 and Theorem 3 admit optimal solutions in canonical form with maximum gap 2, except for the following two cases:

1. In the L-reduction from Ep-Occ-Max-Eq-SAT to MSR-2 and CMSR-2, a strip that is a pair of literal markers has a gap of $q - 1$, which is larger than 2 for $q \geq 4$.
2. In the L-reduction from d-Dimensional-Matching to MSR-$(d+2)$, a strip that is a pair of edge markers may have an arbitrarily large gap if it corresponds to one of many hyper-edges that share a single vertex.

To extend our results in Theorem 1 and Theorem 3 to the corresponding results in Theorem 4, the first case does not matter because we set the parameter q to 2 when deriving the lower bounds for MSR-2 and CMSR-2 from the lower bound for E3-Occ-Max-E2-SAT.

The second case is more problematic, and we have to use a different L-reduction to obtain a slightly weaker asymptotic lower bound for δ-gap-MSR-d. Trevisan [25] showed that Max-IS-Δ is NP-hard to approximate within $\Delta/2^{O(\sqrt{\log \Delta})}$. By Lemma 3, there is an L-reduction from Max-IS-Δ to δ-gap-MSR-$(f(\Delta) + 2)$ with $\alpha\beta = 1$. By the two inequalities in (3), we have $f(\Delta) + 2 = \Theta(\Delta)$. Thus δ-gap-MSR-d is NP-hard to approximate within $d/2^{O(\sqrt{\log d})}$. This completes the proof of Theorem 4.

9 Concluding remarks

A strip of length l has $l - 1$ *adjacencies* between consecutive markers. In general, k strips of total length l have $l - k$ adjacencies. Besides the total strip length, the total number of adjacencies in the strips is also a natural objective function of MSR-d [11]. It can be checked that our L-reductions for MSR-d and δ-gap-MSR-d still work even if the objective function is changed from the total strip length to the total number of adjacencies in the strips. The only effect of this change is that the constant α is halved and correspondingly the constant β is doubled (from $1/2$ to 1). Since the product $\alpha\beta$ is unaffected, Theorem 1 and the second part of Theorem 4 remain valid. For Theorem 2, we can adapt the $2d$-approximation algorithm for maximizing the total strip length to a $(2d + \epsilon)$-approximation algorithm for maximizing the total number of adjacencies in strips, for any constant $\epsilon > 0$. The only change in the algorithm is to enumerate all d-intervals of strip lengths at most $\Theta(1/\epsilon)$, instead of 2 and 3. We note that the small difference between the two objective functions, total length versus total number of adjacencies, has led to difference in the complexities of two other bioinformatics problems [21, 19]: For RNA secondary structure prediction, the problem Maximum Stacking Base Pairs (MSBP) maximizes the total length of helices, and the problem Maximum Base Pair Stackings (MBPS) maximizes the total number of adjacencies in helices. On implicit input of base pairs determined by pair types, MSBP is polynomially solvable, but MBPS is NP-hard and admits a polynomial-time approximation scheme [21]; on explicit input of base pairs, MSBP and MBPS are both NP-hard, and admit constant approximations with factors $5/2$ and $8/3$, respectively [19].

In our Theorem 1 and Theorem 3, we have chosen to display explicit lower bounds for MSR-2 and CMSR-2, despite the fact that they are rather small and unimpressive. As commented by M. Karpinski after the author's ISAAC presentation, it may be possible to improve the lower bound for MSR-2 by an L-reduction from another problem. For example, Berman and Karpinski [8] proved that E3-Occ-Max-E2-SAT is APX-hard to approximate within any constant less than $\frac{464}{463}$ by an L-reduction from Ed-Occ-Ek-LIN-2, and proved that Ed-Occ-Ek-LIN-2 is NP-hard to approximate within some other constant by an L-reduction from yet another problem, and so on. By constructing an L-reduction directly from Ed-Occ-Ek-LIN-2 to MSR-2, say, we might obtain a better lower bound. We were not engaged in such pursuits in this paper. Since satisfiability problems are well-known, we chose an L-reduction from E3-Occ-Max-E2-SAT to MSR-2 for the sake of a gentle presentation, and we made no effort in optimizing the constants.

We proved Theorem 4 by extending our proofs of Theorem 1 and Theorem 3 with minimal modifications. We note that the δ-gap constraint actually makes it easier to prove the APX-hardness of δ-gap-MSR-d and δ-gap-CMSR-d than to prove the APX-hardness of MSR-d and CMSR-d. For example, our E3-Occ-Max-E2-SAT constructions for MSR-2 and CMSR-2 can be much simplified to obtain better approximation lower bounds for δ-gap-MSR-d and δ-gap-CMSR-d. We omit the details and refer to [10] for more results on these restricted variants. On the other hand, the correctness of our reductions does require gaps of at least 2 markers. Thus our proofs do not imply the APX-hardness of 1-gap-MSR-d or 1-gap-CMSR-d. Consistent with our results, Bulteau, Fertin, and Rusu [10] proved that δ-gap-MSR-2 is APX-hard for all $\delta \geq 2$ and is NP-hard for $\delta = 1$.

A curious concept called *paired approximation* was recently introduced by Eppstein [15]. For certain problems on the same input, say Clique and Independent Set on the same graph, sometimes we would be happy to find a good approximation to either one, if not both. Inapproximability results for pairs of problems are often *incompatible*: the hard instances for one problem are disjoint from the hard instances for the other problem. As a result, an approximation algorithm may find a solution to one or the other of two problems on the same input that is better than the known inapproximablity bounds for either individual problem. Note that our inapproximability results for MSR-2 and CMSR-2 are compatible because they are obtained from the same reduction from E3-Occ-Max-E2-SAT. Thus even as a paired approximation problem, (MSR-2, CMSR-2) is still APX-hard. This is the first inapproximability result for a paired approximation problem in bioinformatics.

Postscript. The APX hardness results for MSR-2 and MSR-3 in Theorem 1 was obtained in December 2008. The author was later informed by Binhai Zhu in January 2009 that Lusheng Wang and he had independently and almost simultaneously proved a weaker result that MSR-2 is NP-hard [26].

References

[1] J. Akiyama and V. Chvátal: A short proof of the linear arboricity for cubic graphs, *Bull. Liber. Arts & Sci.*, NMS No. **2** (1981), 1–3.

[2] J. Akiyama, G. Exoo, and F. Harary: Covering and packing in graphs III: cyclic and acyclic invariants, *Mathematica Slovaca*, **30** (1980), 405–417.

[3] J. Akiyama, G. Exoo, and F. Harary: Covering and packing in graphs IV: linear arboricity, *Networks*, **11** (1981), 69–72.

[4] P. Alimonti and V. Kann: Some APX-completeness results for cubic graphs, *Theoretical Computer Science*, **237** (2000), 123–134.

[5] N. Alon: The linear arboricity of graphs, *Israel Journal of Mathematics*, **62** (1988), 311–325.

[6] R. Bar-Yehuda, M.M. Halldórsson, J.(S.) Naor, H. Shachnai, and I. Shapira: Scheduling split intervals, *SIAM Journal on Computing*, **36** (2006), 1–15.

[7] P. Berman and M. Karpinski: On some tighter inapproximability results, in *Proceedings of the 26th International Colloquium on Automata, Languages and Programming (ICALP'99)*, 1999, LNCS 1644, pp. 200–209.

[8] P. Berman and M. Karpinski: Improved approximation lower bounds on small occurrence optimization, *Electronic Colloquium on Computational Complexity*, 2003, Report TR03-008.

[9] L. Bulteau, G. Fertin, M. Jiang, and I. Rusu: Tractablity and approximability of maximal strip recovery, submitted.

[10] L. Bulteau, G. Fertin, and I. Rusu: Maximal strip recovery problem with gaps: hardness and approximation algorithms, in *Proceedings of the 20th International Symposium on Algorithms and Computation (ISAAC'09)*, 2009, LNCS 5878, pp. 710–719.

[11] Z. Chen, B. Fu, M. Jiang, and B. Zhu: On recovering syntenic blocks from comparative maps, *Journal of Combinatorial Optimization*, **18** (2009) 307–318.

[12] M. Chlebík and J. Chlebíková: Complexity of approximating bounded variants of optimization problems, *Theoretical Computer Science*, **354** (2006), 320–338.

[13] V. Choi, C. Zheng, Q. Zhu, and D. Sankoff: Algorithms for the extraction of synteny blocks from comparative maps, in *Proceedings of the 7th International Workshop on Algorithms in Bioinformatics (WABI'07)*, 2007, pp. 277–288.

[14] I. Dinur and S. Safra: On the hardness of approximating minimum vertex cover, *Annals of Mathematics*, **162** (2005), 439–485.

[15] D. Eppstein: Paired approximation problems and incompatible inapproximabilities, in *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'10)*, 2010, pp. 1076–1086.

[16] E. Hazan, S. Safra, and O. Schwartz: On the complexity of approximating k-set packing, *Computational Complexity*, **15** (2006), 20–39.

[17] M. Jiang: Inapproximability of maximal strip recovery, in *Proceedings of the 20th International Symposium on Algorithms and Computation (ISAAC'09)*, 2009, LNCS 5878, pp. 616–625.

[18] M. Jiang: Inapproximability of maximal strip recovery: II, in *Proceedings of the 4th International Frontiers of Algorithmics Workshop (FAW'10)*, 2010, LNCS 6213, pp. 53–64.

[19] M. Jiang: Approximation algorithms for predicting RNA secondary structures with arbitrary pseudoknots, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, **7** (2010), 323–332.

[20] M. Jiang: On the parameterized complexity of some optimization problems related to multiple-interval graphs, in *Proceedings of the 21st Annual Symposium on Combinatorial Pattern Matching (CPM'10)*, 2010, LNCS 6129, pp. 125–137.

[21] R.B. Lyngsø: Complexity of pseudoknot prediction in simple models, in *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP'04)*, 2004, pp. 919–931.

[22] B. Manthey: Non-approximability of weighted multiple sequence alignment for arbitrary metrics, *Information Processing Letters*, **95** (2005), 389–395.

[23] H. Nagashima and K. Yamazaki: Hardness of approximation for non-overlapping local alignments, *Discrete Applied Mathematics*, **137** (2004), 293–309.

[24] C.H. Papadimitriou and M. Yannakakis: Optimization, approximation, and complexity classes, *Journal of Computer and System Sciences*, **43** (1991), 425–440.

[25] L. Trevisan: Non-approximability results for optimization problems on bounded degree instances, in *Proceedings of the 33rd ACM Symposium on Theory of Computing (STOC'01)*, 2001, pp. 453–461.

[26] L. Wang and B. Zhu: On the tractability of maximal strip recovery, in *Proceedings of the 6th Annual Conference on Theory and Applications of Models of Computation (TAMC'09)*, 2009, LNCS 5532, pp. 400–409.

[27] C. Zheng, Q. Zhu, and D. Sankoff: Removing noise and ambiguities from comparative maps in rearrangement analysis, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, **4** (2007), 515–522.

[28] D. Zhu and L. Wang: On the complexity of unsigned translocation distance, *Theoretical Computer Science*, **352** (2006), 322–328.